CHAPTER 4

Forces Affecting the Adoption of Web Services and Other Integration Techniques

Change in any organization can be challenging. This applies to technical and system changes as well. In this chapter, changes in the form of various integration techniques are covered. For each technique, the forces that help and hinder the change are analyzed using a technique called force field analysis. Included in the force field analysis are integration techniques such as adopting enterprise-wide standards, various types of middleware, data warehousing, and message routing.

➤ My first exposure to attempting enterprise integration came many years ago. I worked for a corporation that decided to standardize on what was called "basic business elements" or BBEs. Various departments had different definitions for serial numbers, among other commonly used data elements. This was seen as a bad thing that had to be expensive to the corporation. An analogy was made to a discovery that different sheet metal screws were being used to build equipment in different departments when one type of screw could be used almost universally. Buying, inventorying, and using one type of screw actually saved a considerable amount of money. The reasoning, believe it or not, was that if using one type of sheet metal screw across departments saves money, then using one serial number definition across departments should also save money. The intent was good, but of course, a lot of time and money was spent on defining these BBEs. The analysis seemed to go forever. In fact, the use of BBEs never went beyond the analysis stage.

Force Field Analysis Overview

Force field analysis is a tool that can help us get a perspective on the forces at work when trying to make changes in organizations. This approach to analyzing change was developed by Kurt Lewin.[1] Figure 4.1 illustrates the concepts of this technique. For any particular activity, there is a goal or vision, which is shown by the large arrow at the top of the figure pointing to the right. There are driving and restraining forces that will impact whether this goal or vision can be achieved. Driving forces, which help achieve the goal or vision, are shown as arrows pointing to the right in the same direction as the large arrow at the top. Restraining forces, which hinder goal achievement, are the arrows pointing to the left, in the opposite direction as the large arrow at the top. At some point, driving and restraining forces are in equilibrium. This is illustrated in the figure by the wide vertical line labeled "Status Quo." Driving forces move an organization from the status quo in the direction of the organization's goal or vision. Restraining forces hold back this change from the status quo. These forces can be external or internal to an organization, or external or internal to the individuals in the organization. The relative strength of the driving or restraining forces determines whether change occurs.

Assume, for example, that you want to change a part of a system in an organization. An external organizational driving force could be the opportu-

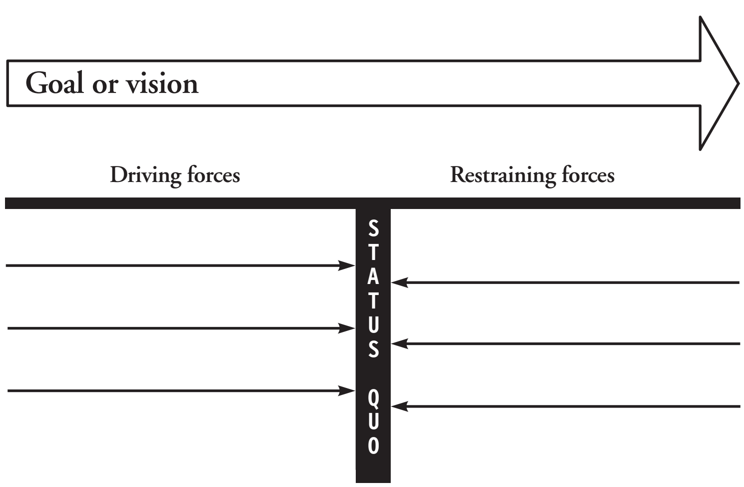

FIGURE 4.1 Force field analysis.

1. Lewin, Kurt, *Field Theory in Social Science,* Harper and Row, New York, 1951.

nity to electronically exchange purchase orders and invoices with a particular customer or supplier. An internal organizational driving force could be a reduction in operating costs. An internal organizational restraining force could be the development cost for making the change. Finally, an internal individual restraining force for some people in the company might be that the change to the system may result in fewer jobs in their part of the organization. Figure 4.2 illustrates this concept.

Of course, there could be many other forces at work than those shown in Figure 4.2. The nature of the driving and restraining forces could also vary by organization even if the organizations were attempting to carry out exactly the same tasks. In fact, they can vary among departments in the same organization.

Essentially, the purpose of this model is to make all the driving and restraining forces visible so that decisions concerning change can be made with the best available information. There are various ways to use this model. If you want to make change more likely, you need to either strengthen the driving forces or weaken the restraining forces. Weakening the restraining forces is sometimes the best approach. Strengthening the driving forces can make the restraining forces get stronger. In Figure 4.2, promoting the electronic interchange capabilities of this change will likely cause more concern about job loss that, in turn, could create various forms of resistance that can effectively scuttle efforts to change from the status quo. So, perhaps it is possible to assure people that jobs will not be lost, thus weakening this restraining force. Of course, this assurance and the resulting reduction in resistance

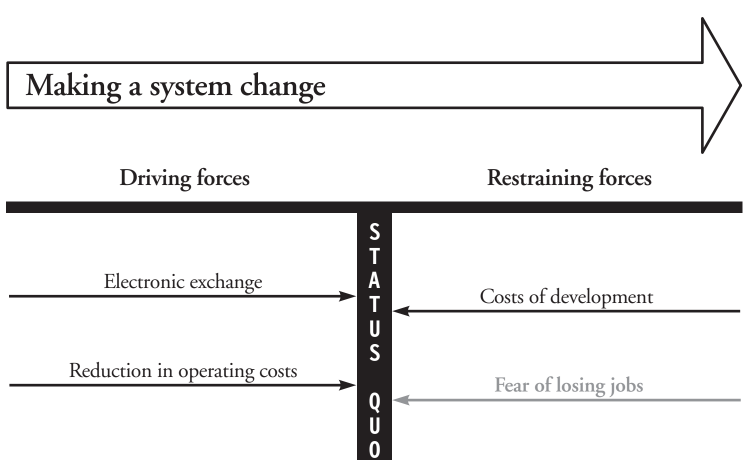

FIGURE 4.2 Force field analysis for making a system change.

would work only if it were actually true that jobs would not be lost. In the figures that follow, weakened restraining forces will be shown as gray arrows to indicate the restraining force is fading away. Figure 4.2, for example, shows that the fear of losing jobs is weakened and less of a concern.

Analysis of Integration Techniques

Common system integration techniques are presented approximately in chronological order. Chronological order allows us to see how, over time, advances in technology and standards have diminished the number of restraining forces, making change more likely to occur. At the end, Web Services will be seen as having the least number of restraining forces compared to any of the other techniques. Using Web Services essentially represents the most recent advance in standards and technology.

The force field analysis presented here will highlight common driving and restraining forces. Every organization will, of course, have additional forces to consider that might be unique to the organization. Also, design issues such as "project scope" or people issues such as "resistance to change" do not appear in these analyses, because they will require special discussion. These issues will be covered in Part II of this book on managing change.

Also, each analysis will include such driving forces as "reduced development time" and "reduced maintenance cost" as universal driving forces.

Analysis of Adopting Enterprise-Wide Standards

The most obvious way to integrate systems within any organization is to establish some type of enterprise-wide standards. This section will cover the forces affecting the adoption of standard data element definitions and the adoption of standard, enterprise software.

Adopting Standard Data Element Definitions

This first analysis shown in Figure 4.3 relates to the vignette that opened this chapter. In the early 1980s, many large organizations were running on custom software and there was very little use of packaged software. At the time, it was believed that there would be opportunities to exchange data more easily, reduce development time, and possibly reduce maintenance costs if all the custom software were to use the same data element definitions. These opportunities are shown as driving forces in Figure 4.3. Restraining forces related to cost offset these driving forces of saving money. Figure 4.3 shows the restrain-

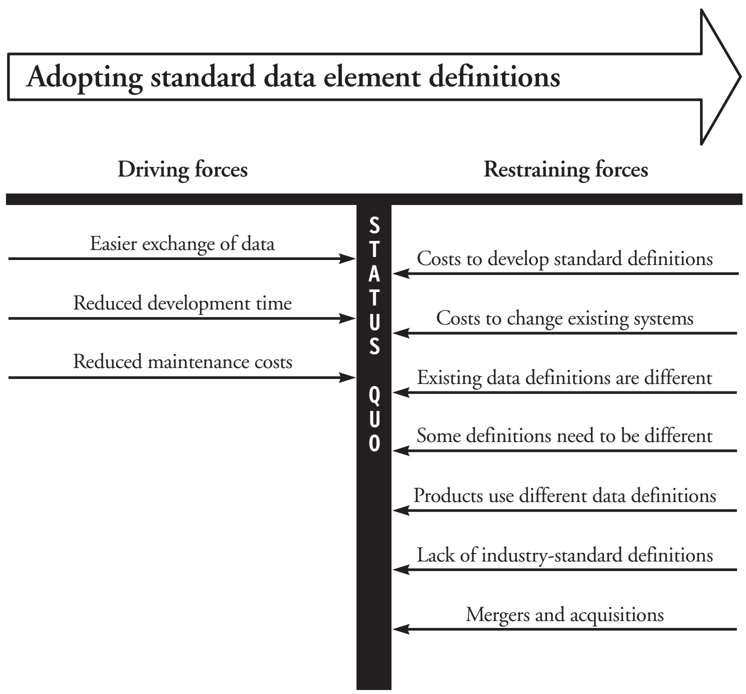

FIGURE 4.3 Force field analysis for adopting standard data element definitions.

ing forces of costs to developing the standard definitions and the costs related to changing the existing systems.

There are additional restraining forces in this figure. In some cases, there were valid reasons that two different systems used different definitions for the same data element. For example, in the vignette opening this chapter, most of the differences related to serial numbers and how they were used among different departments. Also, in the early 1980s, there had been little progress in developing a standard set of data elements for industry. Therefore, the cost of developing a standard set for an organization was quite high because it involved starting with a clean sheet of paper. Even if this effort to change to standard data elements had been successful, the first merger or acquisition would likely cause a problem. The systems used by every other organization would have used different data elements. Finally, as the use of packaged software increased, the definitions used in those products would most likely be incompatible. With enough mergers or acquisitions and use of packaged software, you would be back at the starting point with incompatibility of data definitions.

> *Times have changed since the early 1980s and so have attitudes toward standard data elements. Some industries can see advantages in having standard data elements so that data can easily be interchanged. Another advantage to standard data elements is that they lessen the integration efforts involved in mergers and acquisitions. A sampling of standard vocabularies by industry starts on page 212. So, it would be fair to say that the infrastructure provided by Web Services and XML has given a boost to the standardization efforts.*

Adopting Standard, Enterprise-Wide Software

In some organizations, there is great interest in establishing organization-wide software. This works sometimes. When it does, however, usually it is successful only for a short period. The appeal of adopting standard software is obviously that everyone uses the same software. This means that the entire organization uses the same data definitions, semantics, and formats for exchanging data. Usually this works best for organizations that are small and putting a new set of systems in place. It also works if you are standardizing on non-systems software such as a particular word processor, spreadsheet, or an e-mail system. Nevertheless, standardizing on systems software often runs into problems, too. There are long-term restraining forces, such as mergers and acquisitions that can come into play. Even a new, small organization can acquire another company that uses an entirely different system, and integration problems begin. Figure 4.4 provides the force field analysis for standard, enterprise-wide software.

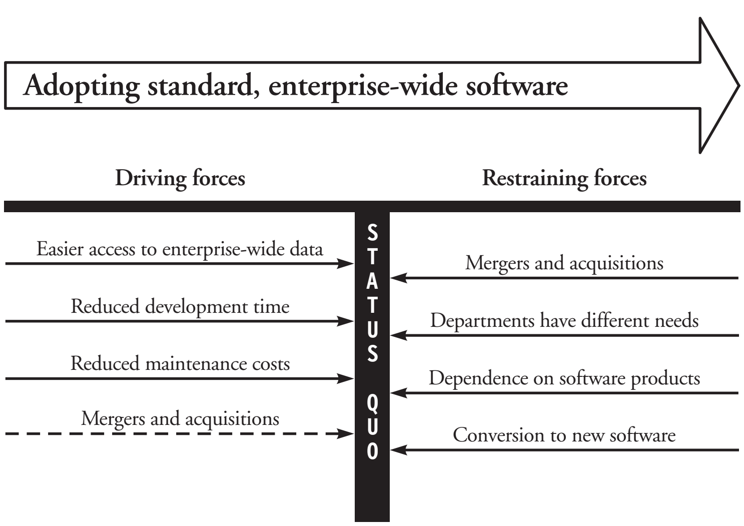

FIGURE 4.4 Force field analysis for adopting standard, enterprise-wide software.

Beside the mergers and acquisitions restraining force, it is common in larger organizations that some departments have different software needs. It is rare that you can find "one size fits all" software. Another downside is that adopting a complete set of software systems from a single vendor makes your organization dependent on that single vendor. As soon as you move away from that vendor's products, you might be back into common integration issues. Finally, for organizations that have existing systems, adopting standard software can mean a mass conversion to the new software. This often is problematic and should be seen as a restraining force.

Note that none of the restraining forces in this figure are shown in gray. This means that they will not diminish over time and will remain restraining forces going forward into the foreseeable future.

There is, however, increasing recognition among software vendors that it is in their interest to create plug-compatible software components that can be used in assembling a service-oriented architecture. This plug-compatibility will be achieved using Web Services. Using plug-compatible software will make it easier to "mix and match" vendor products when developing or modifying an enterprise architecture.

Of course, every example has a counter example. There are some industries where mergers and acquisitions are commonplace. You will see organizations in those industries adopting common, industry-wide software packages so that it will be easier for one organization to be acquired or merged with another organization. So, mergers and acquisitions can also be a driving force. This is represented in Figure 4.4 with a dashed line. Although, I have not seen any empirical data on this, my experience is that this is the exception rather than the rule. That is the reason for the dashed line because it is likely to not apply to all industries. Similarly, mergers and acquisitions could be a driving force for current efforts on developing industry-wide vocabularies for Web Services.[2]

Analysis of Middleware Integration

Middleware hides the complexity of the communication between two or more systems or services. This simplifies the development of those systems and services and isolates the complexity of the communication between them. The different systems or services can be on the same hardware or on different hardware. Figure 4.5 shows the basic middleware architecture.

2. See page 212 for a sampling of industry-wide vocabularies.

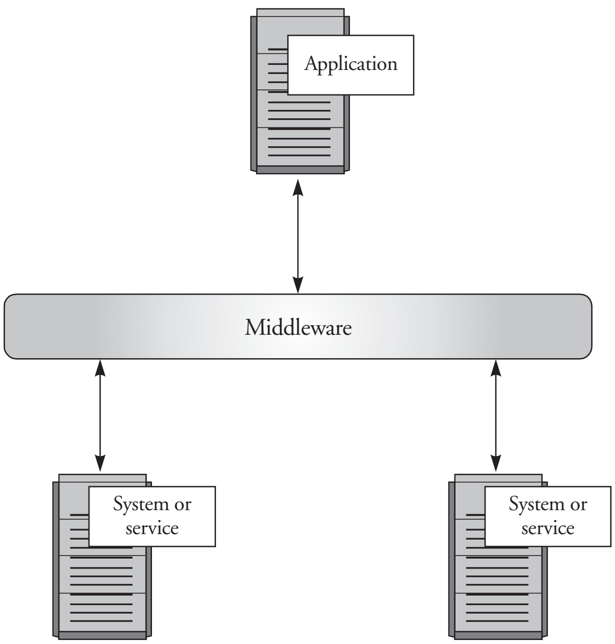

FIGURE 4.5 Basic middleware architecture.

In Figure 4.5, the different systems and services each are running on different machines. These machines could be in the same location or different locations for the same organization. Alternatively, all three machines could belong to different organizations.

There has been significant work on middleware over the years. Some examples of middleware include:

- **Transaction processing (TP) monitors.** A TP monitor ensures that transactions process completely or the appropriate action is taken if an error occurs. They often employ load balancing because a transaction may be forwarded to any of several servers.
- **Remote Procedure Call (RPC).** An RPC allows execution of program logic on a remote system by calling a local routine.
- **Message-Oriented Middleware (MOM).** MOM provides program-to-program data exchange.
- **Object Request Broker (ORB**). An ORB allows a system to request a service without knowing anything about what servers are available. The request is forwarded to the appropriate services with the results of the request returned to the requesting system.

This section will examine adopting two forms of existing middleware and compare that to adopting Web Services as middleware. The two forms commonly used are the Object Management Group's Common Object Request Broker Architecture (CORBA) specification and Microsoft's Distributed Common Object Model (DCOM).

Adopting CORBA and DCOM

CORBA and DCOM are middleware that provide a means for applications to communicate with each other. CORBA is a set of specifications developed through the Object Management Group (see page 222). Implementations of CORBA are referred to as *ORBs* or *object request brokers.* DCOM comes from Microsoft (see page 223). CORBA and DCOM are interoperable through CORBA/DCOM bridges. Both CORBA and DCOM represent reasonably mature technology for creating interoperable, networked applications. Figure 4.6 shows that providing interoperable, networked applications are a driving force for adopting CORBA, DCOM, or in some cases both.

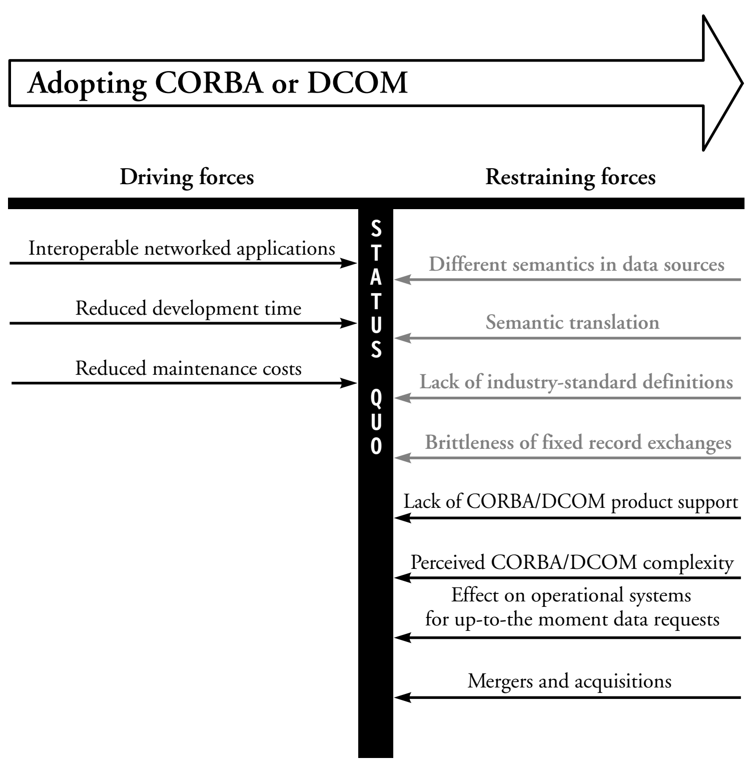

FIGURE 4.6 Force field analysis for adopting CORBA or DCOM.

CORBA and DCOM, however, are the means to get data from one place to another. There are no specific requirements for the format of the data transmitted in the messages, which means that the transmitted data might not be workable when it arrives at its destination. The restraining forces related to data exist with either CORBA or DCOM. These are:

- Different semantics in data sources
- Semantic translation
- Lack of industry-standard definitions

Advances in industry standards for XML,[3] EDI,[4] and Web Services (see page 205) will mitigate all these restraining forces, which is why they are shown as gray arrows in Figure 4.6. In fact, using XML (see page 209) with CORBA or DCOM makes for a more flexible system because of the tagged record structure of XML.[5] This would also mitigate the restraining force related to the brittleness of fixed record formats.

There is a perception in the industry that neither CORBA nor DCOM are that widely adopted and that using one or the other—or both—is too complex for many programmers. Whether the perceived lack of industry adoption or inherent complexity is actually true is irrelevant at this point. These perceptions are seen as restraining forces. In fact, they should be seen as the most significant restraining forces. Web Services have just the opposite perception—they are seen as easy to adopt widely by industry and easy for most programmers to use. Perception in this case might well be the reality.

The very nature of creating interoperable, networked resources means that there could be a negative impact on operational systems when requests come in through CORBA or DCOM. Many operational systems have not been designed to receive ad hoc or unexpected processing requests. These requests sometimes can have a negative impact on the performance of those systems. So, the effect on operations systems can be a restraining force, but should be expected if up-to-the-moment processing is needed (see page 161).

Finally, mergers and acquisitions could be an issue because it is entirely possible that the acquired systems do not use either CORBA or DCOM, and it is not a trivial task to retrofit systems to use either technology.

---

3. A sample of industry-specific XML vocabularies are listed starting on page 212.

4. The ebXML Registry is an example for EDI. It is discussed on page 208.

5. For an explanation of the tagged record structure of XML and the brittleness of fixed record formats, see page 26.

Adopting Web Services

Using Web Services is the missing piece in the puzzle of how to create interoperable systems and services and benefits from the development of CORBA and DCOM. Web Services use of both XML and HTTP[6] on the Internet greatly reduces restraining forces that existed with preceding technologies. The perceived simplicity of Web Services has created a stampede of vendors incorporating Web Services into their products and services. Figure 4.7 shows the driving and restraining forces for adopting Web Services.

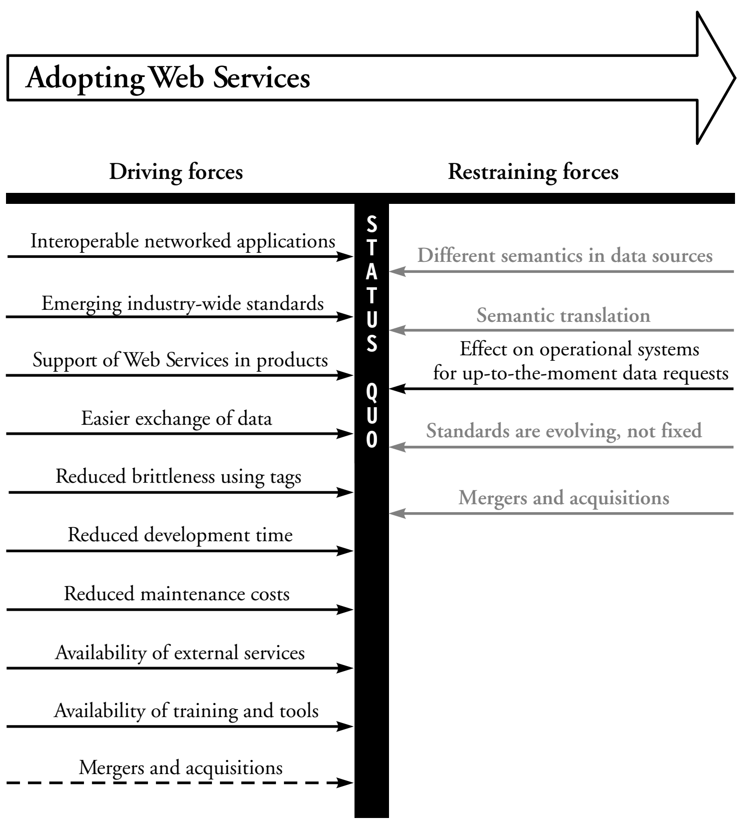

FIGURE 4.7 Force field analysis for adopting Web Services.

6. One reason other protocols have failed to gain wide acceptance is the desire to use HTTP. The CORBA Internet Inter-ORB Protocol (IIOP) is an example of a protocol that did not gain wide acceptance, in part for that reason.

In addition to the same driving force of interoperable, networked applications shown for CORBA and DCOM, the driving forces in Figure 4.7 now include a driving force related to XML's tagged format: reduced brittleness using tags. The explosive growth of Web Services being used in products will also result in a significant growth of external services that can be used by organizations of any size. The nature of services available will be covered in Chapter 5. Similarly, there is an abundance of training and tools becoming available on Web Services.

The restraining forces are similar to those we have seen with earlier integration techniques. These include:

- Different semantics in data sources
- Semantic translation

There is a tremendous amount of work in various standards organizations to simplify the semantics and standardize the semantic translation (see page 212). Those standards are evolving, which is why it is seen as a restraining force. Nevertheless, as time goes on, these restraining forces will weaken.

Finally, mergers and acquisitions, which appeared as a restraining force for CORBA and DCOM, is shown in Figure 4.7 as a weakening restraining force for the adoption of Web service. The broad adoption of Web Services by product vendors over time increases the likelihood that an acquired company will be able to work with Web Services. Mergers and acquisitions also appear as a driving force. This is for those industries where mergers and acquisitions are commonplace. Mergers and acquisitions could, for example, be a driving force for current efforts on developing industry-wide vocabularies for Web Services (see page 212). The reason for the dashed line in Figure 4.7 is because this driving force is likely to not apply to all industries.

Just as with CORBA and DCOM, you are left with the possible impact on operational systems, but this should be expected if up-to-the-moment processing is needed. As mentioned in that discussion many operational systems have not been designed to receive ad hoc or unexpected processing requests. These requests sometimes can have a negative impact on the performance of those systems. So, the effect on operations systems can be a restraining force, but should be expected if up-to-the-moment processing is needed.

Adopting Web Services Does Not Mean Abandoning CORBA or DCOM

For organizations that have invested heavily in CORBA or DCOM, the adoption of Web Services does not mean you need to abandon CORBA or DCOM and convert existing systems. Recall that the middleware hides the complexity of the communication between two more systems or services. That hiding of complexity allows existing systems to participate in Web Services by altering the way the middleware works. Figure 4.8 is a high-level view of how Web Services can work with both CORBA and DCOM with each of the systems or services operating in a different organization.[7]

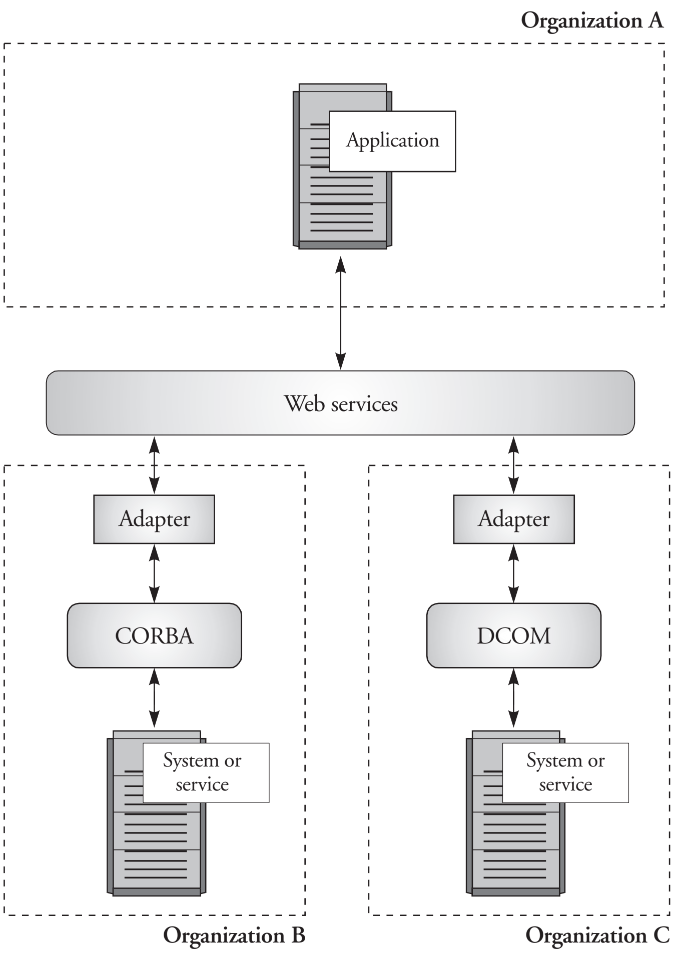

FIGURE 4.8 Web Services interoperating with CORBA and DCOM.

7. Of course, a similar figure could be drawn for other forms of middleware. Web Services is helping to drive the interoperability of all forms of middleware.

Analysis of Additional Components Used for Integration in a Service-Oriented Architecture

Two additional techniques of integration can be used in a service-oriented architecture. They are data warehousing and message routing. This section discusses those techniques and how they can be integrated with middleware such as CORBA, DCOM, and Web Services. Also, this section will show how efforts around Web Services will enhance both data warehousing and message routing.

Analysis of Data Warehousing

One of the oldest and most successful ways to integrate data from multiple systems is to extract that data from existing systems and load it into a single, central location to form an enterprise data warehouse (EDW). Using an EDW can be complementary to using CORBA, DCOM, or Web Services. The analysis for this approach is shown in Figure 4.9.

In this figure, the easier exchange of data as a driving force is replaced with easier access to enterprise-wide data. This data is loaded from existing systems using various techniques that extract, transform, and load the data in the EDW. Using extract, transform, and load (ETL) techniques means there is usually less impact on operational systems because the extracts of data from these systems could be done at a time convenient for the operational system. This lowered impact on operational systems is a significant driving force. Easier access to enterprise-wide data also allows the use of business intelligence (BI) software to find patterns or new business opportunities based on a wealth of data that could be stored in an EDW (see page 220).

Most of the restraining forces are issues with the semantics or meaning of the data and the standardization of data definitions. Not surprisingly, these issues are similar to those involved with attempts at adopting standard data elements when existing data definitions are different. In this figure, the semantic translation is added to show the need to transform data can itself be a restraining force. Over time, however, these restraining forces have become weaker for two reasons:

1. **A subset of our industry is devoted to the development of ETL software.** This software generally simplifies the development the data extractions from existing systems, any semantic translation or transformation, and the loading of the data into the EDW.
2. **More industry standards have become available.** Initially with efforts related to Electronic Data Interchange (EDI) and more recently with Web Services.

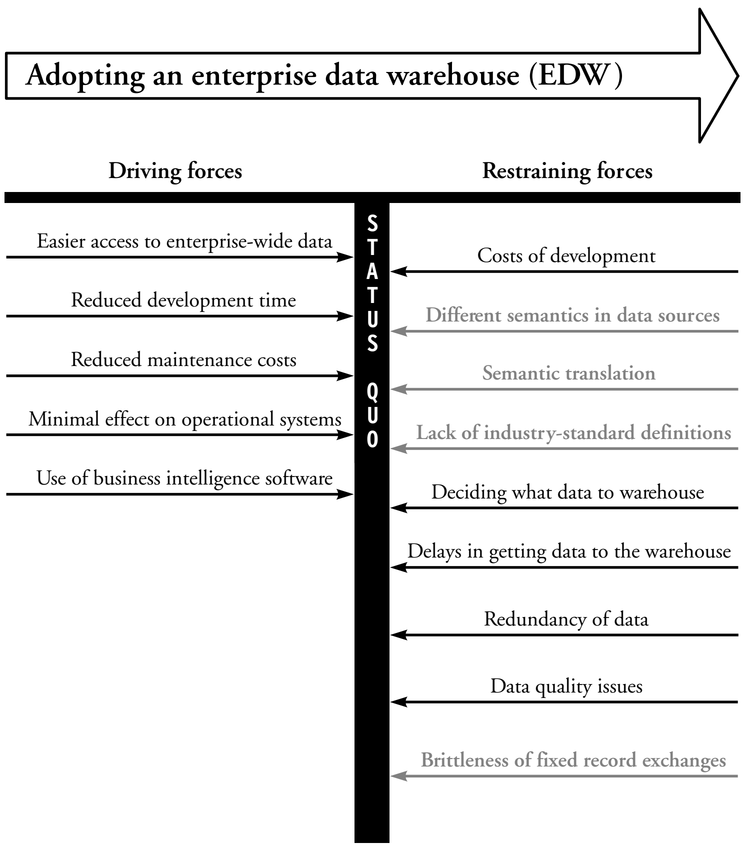

FIGURE 4.9 Force field analysis for adopting an enterprise data warehouse.

Additional restraining forces include problems related to what data to store in the EDW, and the delay or latency of getting data into the EDW. The issue of what data should be stored in an EDW will likely always remain a restraining force. The strength of this restraining force will vary by organization. The delay or latency of data is the result of performing data extracts at times convenient to the operational systems.[8] Consequently, the very latest data is not always available in the EDW. To some organizations, this is no problem. Others, however, may find this a significant restraining force.

Redundancy of data also can be seen as a restraining force. Whenever data exists in more than one location, it is possible that the data will have different

8. For some organizations, this can be a certain time of day. For others who cannot stop their operational systems, it may be necessary to provide small data extracts throughout the day.

values for various reasons. This could result, in part, from the latency of data mentioned earlier. For example, the value of an account balance may be updated by the operational system but not forwarded to the EDW until some later date. At a given point in time, you could see two different values for the same account when looking at the EDW and the operational system. Often, the way to resolve redundancy issues is to create a master database that all systems should use.

Data quality issues are potentially a restraining force, because much depends on the quality of the data available. If data to be stored in the EDW is lacking in quality, there are options available for improving its quality. Changes could be made to improve data quality at the time it is entered. For existing data, the quality could be improved at the source. If that is not possible, the ETL software used to load data could be used to improve the quality of the data. Sometimes this is called *data cleansing* (see page 222). This, of course, assumes the quality can be improved in some way that lends itself to programming. Data quality is a significant topic and you are encouraged to study it further if this is potentially a restraining force for your organization.

Finally, the brittleness of fixed record exchanges is a maintenance issue (see page 26). If the EDW is changed in some way, it could create a need to change some or all the ETL programs. Because of the nature of fixed record exchanges, there is always a chance that not all ETL programs were updated and the wrong data is extracted. As a result, the transform and load portion could fail or the wrong portion of the record could be inappropriately transformed and loaded into the EDW resulting in essentially a corrupted EDW. This brittleness problem is being addressed by the tagged record structure of XML (see page 26). The tagged structure significantly reduces that chance of corrupting the data in the EDW and also presents the opportunity to reduce maintenance costs related to ETL programs (see page 224). So, as a restraining force, the brittleness of fixed records will be reduced. Many of the restraining forces will be reduced because of efforts related to industry standards, XML, and Web Services as represented by the gray arrows in Figure 4.9.

Use of an EDW can be coupled with the use of CORBA, DCOM, or Web Services as shown in Figure 4.10.

Analysis of Application or Message Routing

Often when integrating systems, there is also a need to propagate data among internal systems. For example, it is believable that if a customer's address were changed in one internal system, you would want that change to appear as soon as possible in other internal systems.

If each internal system were directly connected to the other internal systems shown at the bottom of Figure 2.1, you could have up to 15 intercon-

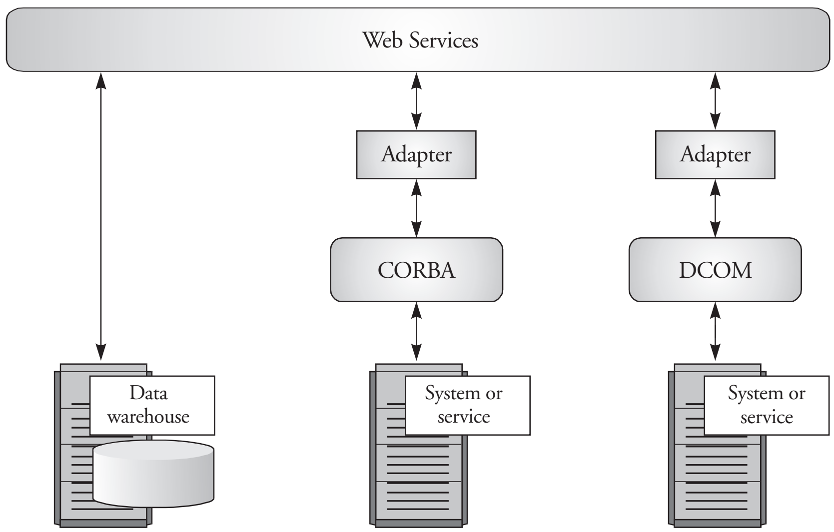

FIGURE 4.10 Enterprise data warehouse co-existing with Web Services, CORBA, and DCOM.

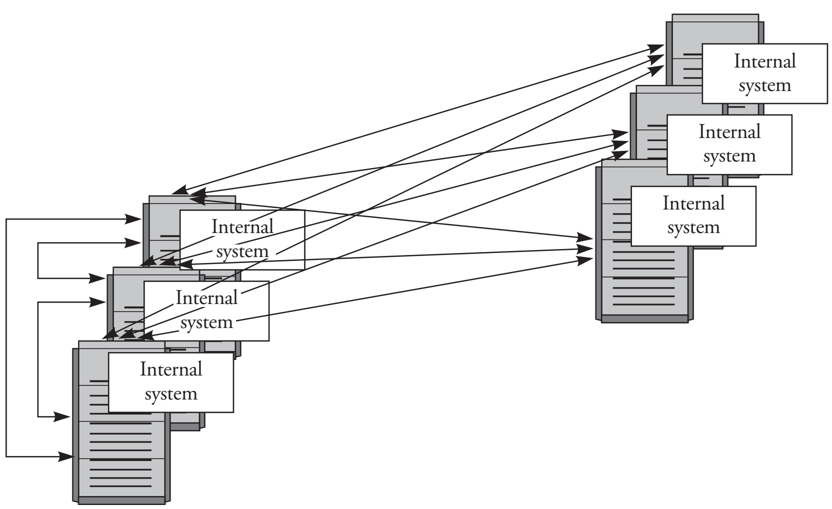

FIGURE 4.11 Some possible interconnections for internal systems.

nections. Some of those connections are shown in Figure 4.11. Of course, if you need to propagate an update, such as a customer address, to multiple systems, you could end up in the situation shown in Figure 4.11. In this situation, every system potentially may need to communicate to other internal systems.

Architecturally, a good solution to this problem is to add a *message router* to internal systems as shown in Figure 4.12. Such routers have been around for some time. They are also known as *application routers.* This message router, however, would be based on Web Services. Also, a router could know which of the other internal systems needs to receive a certain type of updates. The individual internal systems would not need to know who receives such updates. As a result, the number of interconnections is reduced as shown in Figure 4.12.

A message router usually needs to transform the data in some way to match the format of the data expected by the receiving system. Figure 4.13 shows examples of such transformations. Internal system A at the left is sending data in tagged XML format. Internal system B at the right expects a tagged XML format, but expects the tags to be different. For example, instead of the tag `<name>` in system A, system B expects the data to be tagged with `<customer>`. The tags for phone and postal code data also are different. The message tag itself varies as well. At the left, the tag is `<GetCustomerInfo-Response>` and at the right, the tag is `<GetCustomerAccount>`. Finally, System C expects a fixed record format. This fixed format is shown at the bottom of Figure 4.13. The message router needs to "know" how to make these transformations.

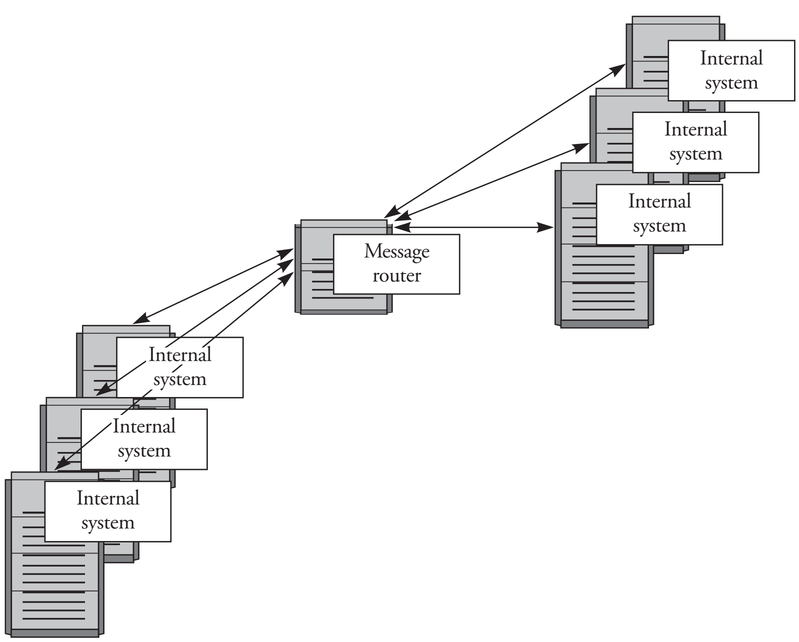

FIGURE 4.12 Interconnections when using a message router.

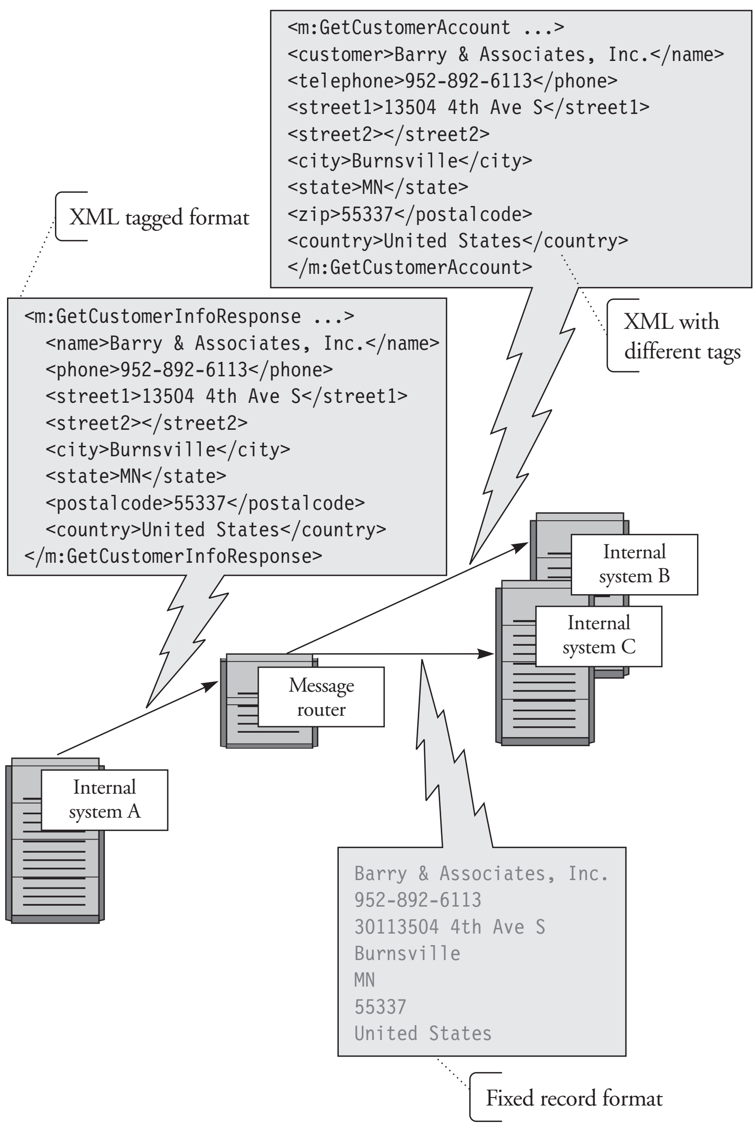

FIGURE 4.13 Transformations in a message router.

In some ways, a message router is the flip side of EDW. Message routing disperses data where EDW collects data. Nevertheless, the analyses of the two approaches are similar. The analysis for adopting a message router is shown in Figure 4.14. In this figure, a driving force is consistent enterprise-wide data in all applications. This means that customer data, for example, would be the

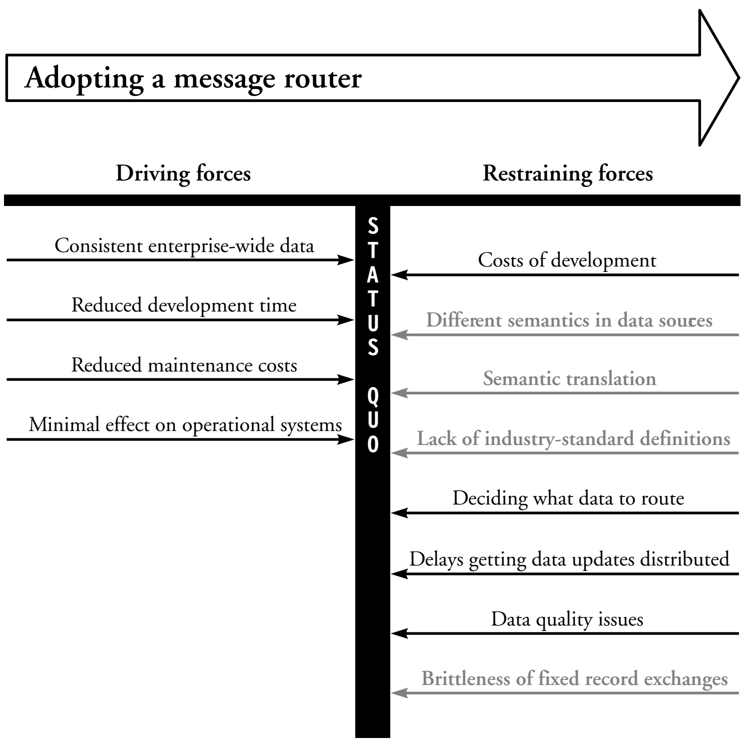

FIGURE 4.14 Force field analysis for adopting a message router.

same no matter what system used or managed that data. The impact on operational systems is also lowered since any one system only needs to communicate with the message router and not all the other internal systems.

Most of the restraining forces are the issues with the semantics or meaning of the data and the standardization of data definitions that have been discussed earlier. Message routing, like EDW, needs to deal with semantic translation and this is shown as a restraining force. Over time, however, these restraining forces have become weaker as more industry standards become available.

Additional restraining forces include problems related to what data to route and the delay or latency of getting data updates distributed to various internal systems. The issues of what data to route and the delay of getting data updates distributed will likely always remain a restraining force.

Data quality issues similar to EDW can occur with message routing. Obviously, it can be potentially disastrous to route poor quality data. With message routing, however, you do not have the option of data cleansing used in conjunction with ETL software. The quality of data needs to be improved at the source for existing data and at the point of entry for new data.

Finally, the brittleness of fixed record exchanges is a maintenance issue.[9] If the format of the record going to the message router is changed in some way, it could create a problem. Because of the nature of fixed record exchanges, there is always a chance that the wrong data is routed. This brittleness problem is being addressed by the tagged record structure of XML. The tagged structure significantly reduces that chance of corrupting the data in the message router and presents the opportunity to reduce maintenance costs related to message routing programs. So, as a restraining force, the brittleness of fixed records will be reduced over time.

Web Services adapters for packaged software provided by vendors will also reduce costs of development. The adapters allow Web Services connections with internally developed systems or packaged software. The arrow depicting that restraining force of development cost, however, is not shown as gray since there are still other significant development costs related to a message router. Nevertheless, many of the restraining forces will be reduced because of efforts related to industry standards, XML, and Web Services as represented by the gray arrows in Figure 4.14.

When a message router uses Web Services, it is depicted in this book as shown in Figure 4.15. Essentially, all the same data paths shown in Figures 4.11 and 4.12 are retained. For the purposes of creating a readable figure, only one line is shown connecting to the message router. This figure also shows two of the internal systems using an adapter. This is meant to represent that some internal systems may need adapters that are not part of the internal system. These adapters could be written in-house or purchased from third-party software vendors.

Just like EDW, message routing can also work with existing middleware solutions such as CORBA and DCOM. This is shown in Figure 4.16. The message router would have connections to Web Services that, in turn, would connect to CORBA and DCOM. As explained earlier, the message router would "know" what data should be routed and when, in some cases, the identifying tag might need to be changed for the receiver of the data.

Putting All the Integration Techniques Together in a Service-Oriented Architecture

Middleware integration, data warehousing, and message routing all work together in a service-oriented architecture. Figure 4.17 shows all these technologies as components of a service-oriented architecture. This is essentially a

9. For an explanation of the brittleness of fixed formats, see page 26.

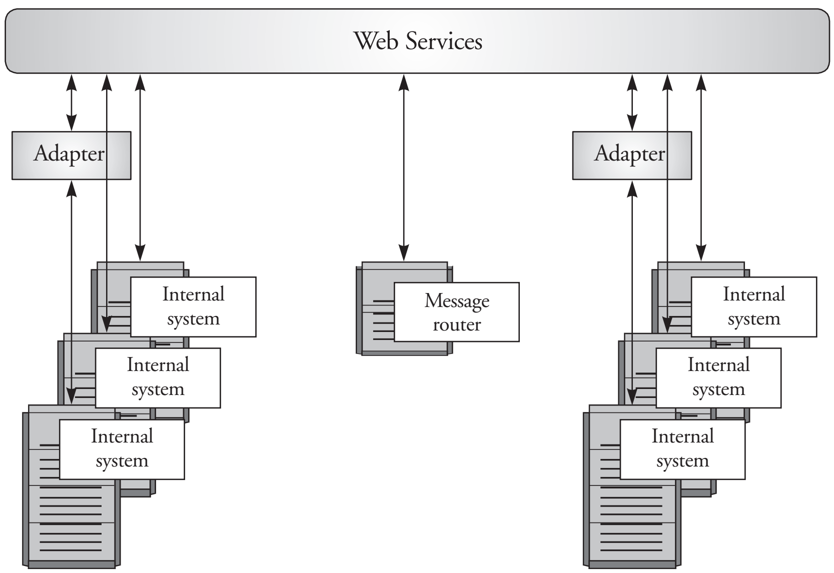

FIGURE 4.15 A message router using Web Services.

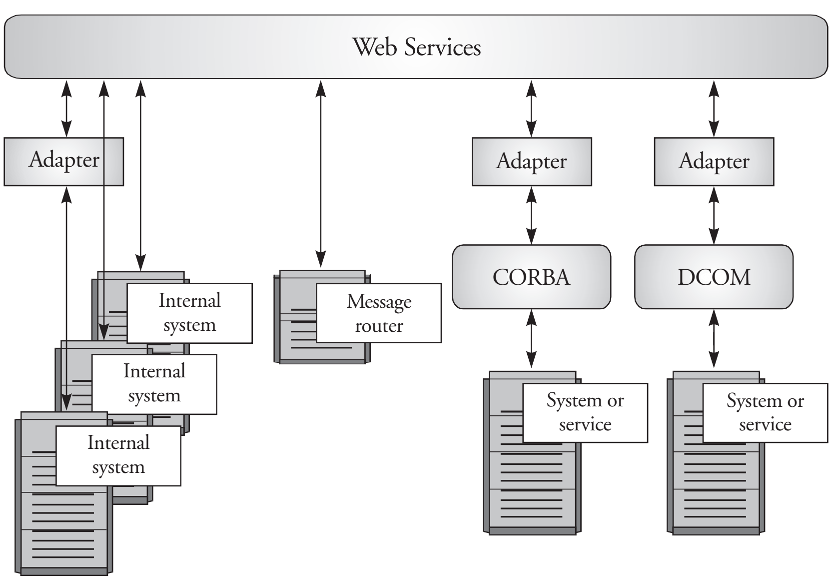

FIGURE 4.16 Message router used with Web Services, CORBA, and DCOM.

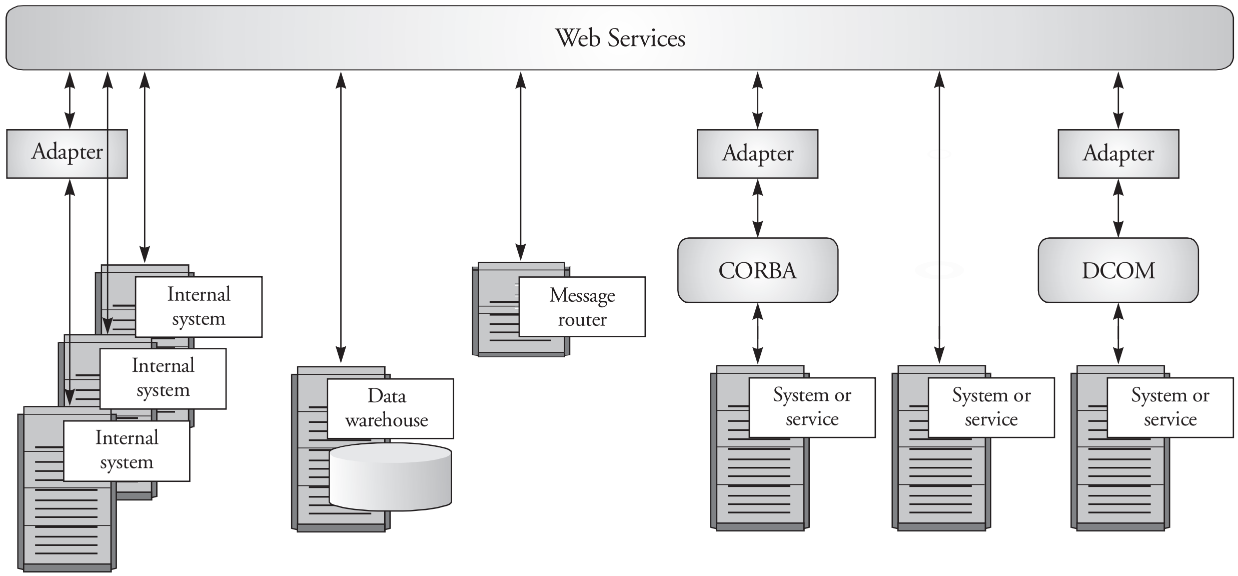

FIGURE 4.17 Multiple components of a service-oriented architecture.

more detailed diagram of C. R.'s organization, which was described in Chapter 2. In Figure 2.1 at the bottom, you can see the internal systems in C. R.'s organization along with the online repository. The three internal systems at the left in Figure 2.1 relate to the three systems at the left in Figure 4.17; this time we add the detail of an adapter for one of the internal systems. The three internal systems at the right in Figure 2.1 relate to the three systems at the right in Figure 4.17; this time one system is CORBA based, a second uses Web Services directly, and a third is DCOM-based. The online repository at the bottom, middle in Figure 2.1 is shown as a message router and a data warehouse in Figure 4.17. Finally, Figure 4.17 shows that C. R.'s organization uses Web Services as the means of interconnecting systems. This detail is not shown in Figure 2.1.

Summary

This chapter used force field analysis to show how various forces drive or restrain the adoption of integration techniques. The discussion showed that using Web Services reduces barriers to integration:

- There are far fewer restraining forces for adopting Web Services than for adopting enterprise-wide standard data element definitions or enterprise-wide software.
- Within middleware, the restraining forces for adopting CORBA or DCOM are much stronger than restraining forces for adopting Web Services. Also, adopting Web Services has more driving forces.
- The standardization efforts related to the use of XML by Web Services are assisting other integration techniques. This was shown in the weakening restraining forces for adopting an enterprise data warehouse and/or a message router.
- Because the use of Web Services does not require abandoning CORBA or DCOM, there are no required changes to existing CORBA or DCOM-based systems. This further reduces barriers to the adoption of Web Services as part of an integration strategy.

The next chapter will cover how the impact of Web Services will grow over time. It uses one of the most significant integration efforts of all time, Electronic Data Interchange, to illustrate the growing impact of Web Services.